L23: Entry 3 of 30 File: USPT Aug 7, 2001

DOCUMENT-IDENTIFIER: US 6272580 B1
TITLE: Apparatus and method for dynamically elevating a lower level bus master to an upper level bus master within a multi-level arbitration system

Abstract Text (1):
A computer system, bus interface unit, and method are provided to allocate requests to a shared bus within the computer system. The bus interface unit includes an arbiter which employs a multi-level, round-robin arbitration protocol. Configuration registers are programmed during boot-up of the computer system by assigning a subset of peripheral devices, bus agents, requesters, or bus masters to either a high priority ring or a low priority ring, if two levels of arbitration are used. The status of a low priority device can be elevated to equal priority with a high priority device by assigning the low priority device to a high priority port within the high priority ring if certain circumstances occur. Namely, if data transfers to or from the low priority device are terminated, then the low priority device will be promoted to a high priority device so that it need not wait until after the all high priority device requests have been polled. Instead, the elevated low priority device is placed on the same level of priority as the high priority devices so that its request can be readily serviced and the transaction completed during a data transfer retry operation.

Brief Summary Text (9):
It would be desirable to introduce a multi-level arbitration mechanism which can assign higher priorities to low latency agents and lower priority to high latency agents. As defined henceforth, the shared resource is a peripheral bus, and the agents are peripheral devices which are operably connected to that bus. The peripheral bus includes a peripheral component interface ("PCI") bus and peripheral devices include all devices either directly or indirectly coupled to, e.g., the PCI bus. The peripheral devices may therefore entail devices such as the microprocessor which is connected to the peripheral bus through a bus interface unit.

Brief Summary Text (12):
The problems outlined above are in large part solved by an improved computer system arbitration protocol. The improved protocol entails a multi-level round-robin arbitration scheme which assigns various agents or peripheral devices to either the high priority arbitration ring or the low priority arbitration ring (if two rings are used) during boot-up of the computer system. A configuration register is programmed during boot-up to retain values indicating the high or low priority status of each peripheral device. In addition, the configuration register is programmed to ensure an initial priority within the high and low priority rings. For example, CPU requests are given a higher priority than requests from an Industry Standard Architecture ("ISA") peripheral device during the initial arbitration sequence of either the high or low priority arbitration rings. The configuration register therefore assigns specific peripheral devices to specific ports on either the high or low priority rings.

Brief Summary Text (14):
The high priority ring is designed so that peripheral devices connected to high priority ports will be given priority over any peripheral devices connected to low priority ports within the low priority ring. As such, the devices on the high

priority ring will be polled first, with one port reserved on the high priority ring for transitioning to the low priority ring. It is not until after all high priority devices between the high priority pointer and the low priority ring ported on one high priority port will the first low priority device be polled. If the high priority ring involves M number of potential masters then, depending on where the high priority pointer is located, up to M arbitration cycles are needed before priority is given to a low priority device. Once the low priority device access has been completed, then both the high and low priority pointers are advanced one port location. If there are N number of low priority devices assigned to low priority ports then, in a worst case scenario, a low priority device will get access to the bus once every N times M arbitration cycles. Almost all peripheral buses employ a sequence of granting mastership before transferring data. The arbitration cycle must therefore precede a data transfer cycle. Unfortunately, instances may arise during data transfer which can temporarily disrupt that transfer. A procedure known as "retry" is often used as an attempt to re-arbitrate for the bus and, therefore, complete the previously interrupted data transfer. If data transfer to or from a low priority device is interrupted, the low priority device would have to wait up to N times M arbitration cycles before it can regain mastership of the bus during its retry operation. This delay is in most instances unacceptable since data transfer of the low priority device is often a prerequisite for subsequent data transfers of high priority devices. Accordingly, one benefit hereof is to implement a mechanism and methodology for temporarily elevating a low priority device to the high priority ring if its data transfer should become interrupted. In this fashion, the low priority device is given equal status with high priority devices and will be granted access to the bus ahead of its previous counterpart low priority devices. However, since the high priority pointer is incremented one port location, the elevated low priority device must wait a maximum of M times before it is again granted access. This is helpful to ensure undue thrashing of ineffective requests from a locked device resource.

Brief Summary Text (16):
A temporary elevation register is used to note which low priority peripheral devices are elevated to a high priority status. Bit locations within the temporary elevation register correspond to similar bit locations within the configuration register. Values within the corresponding bit locations are logically combined to produce either a "1 " or a "0" value outcome. Those outcomes are stored in a register of the same size as the configuration register and the temporary elevation register. That register is then polled by looking for 1s and 0s depending on whether the high priority ring or low priority ring, respectively, is queried. All devices corresponding to a 1 value within the register subsequent to the high priority pointer will be given priority based on their relationship or proximity subsequent to that pointer. All devices having a 0 value within the register are given priority if high priority devices between the high priority pointer and the high priority slot occupied by the low priority ring do not issue a bus request. In this instance, the prioritization will transition to devices on the low priority ring (i.e., devices having a 0 value within the outcome register).

Brief Summary Text (17):
Broadly speaking, a bus arbiter is provided. The bus arbiter comprises a low priority arbitration ring having a plurality of low priority ports assigned to respective low priority peripheral devices. The arbiter further includes a high priority arbitration ring having a plurality of high priority ports assigned to respective high priority peripheral devices as well as one of the low priority peripheral devices elevated from the low priority ports. The low priority peripheral device is elevated during times in which data transfers to and from that low priority peripheral device are interrupted.

Brief Summary Text (18):
According to another embodiment, a computer is provided. A computer includes a microprocessor and a plurality of high and low priority peripheral devices. At

least one of the peripheral devices is coupled to a printed circuit board separate from another printed circuit board on which the microprocessor resides. Accordingly, at least one of the peripheral devices may be arranged upon a card or coupled to a card separate from a motherboard on which the microprocessor is either arranged or directly coupled. An arbiter is provided within the computer and coupled to a peripheral bus on which the plurality of high and low priority peripheral devices are directly or indirectly coupled. The arbiter includes a high priority arbitration ring having a plurality of high priority ports assigned to the plurality of high priority peripheral devices. The arbiter further includes a low priority arbitration ring having a plurality of low priority ports assigned to the plurality of low priority peripheral devices. One of the low priority peripheral devices is assigned to one of the plurality of high priority ports and is granted mastership on the peripheral bus before any of the other low priority peripheral devices whenever a data transfer to or from the one of the plurality of low priority peripheral devices is interrupted or retried. Accordingly, if data transfer is being retried to the elevated low priority peripheral devices, that device is granted mastership over other low priority peripheral devices and is of equal priority status to that of the high priority peripheral devices coupled to the high priority ports.

Brief Summary Text (19):
According to yet another embodiment, a method is provided for granting accesses to a peripheral bus. The method includes configuring a first register to assign a high priority grouping of peripheral devices to a high priority ring of an arbiter and a low priority grouping of peripheral devices to a low priority ring of the arbiter. A second register is configured to assign one of the low priority peripheral devices to the high priority ring if data transfer to or from said one of the low priority peripheral devices within a low priority grouping is interrupted. As such, the first register is a configuration register which is presented values during configuration of the computer system, whereas the second register is a temporary elevation register assigned values whenever data transfers are interrupted. Access to the elevated peripheral device within the low priority grouping is granted before granting access to any of the remaining peripheral devices within the low priority grouping.

Detailed Description Text (13):
During boot-up from BIOS a configuration register possibly associated with the southbridge chipset will contain bit values indicating whether an associated peripheral device will be grouped within the high priority ports or the low priority ports. A programmed 1 value will indicate apportionment to the high priority ring, and a programmed 0 value will indicate apportionment to the low priority ring. The order of priority within each ring is fixed during boot-up. That is, a CPU request is given higher priority than an ISA request, and the ISA request is given a higher priority than requests to a PCI input/output device. Furthermore, any requests to a PCI input/output device is given a higher priority than a request to the sub-function controller deemed to monitor accesses upon either the USB, I.sup.2 C, or the IDE buses. For example, bus master M1 can be deemed the CPU device, M2 the ISA device, M3 the first PCI I/O, etc. until M10 noted as the sub-function device. During boot-up, M1 is placed in a higher priority position than M2 within the high priority ring 40, and M3 is placed at a higher priority position than M5. This procedure is continued for assigning priority among bus masters during boot-up.

Detailed Description Text (14):
A high priority pointer may initially be presented to the CPU master after boot-up. However, while the PCI bus is undergoing reset, the grant signals are tri-stated and the initial park state will remain upon the southbridge. It is not until after the first request that the arbiter will then be parked upon the CPU and, e.g., M1.

Detailed Description Text (23):

FIG. 7 illustrates one exemplary implementation by which high priority and low priority rings can be formulated and pointers directed to those corresponding rings. More specifically, an elevation register 86 is shown in combination with configuration register 84. Elevation register 86 includes bit positions equal in number to those of configuration register 84, and each bit position is assigned to a particular bus agent/master. Given the example shown in FIG. 6, configuration register 84 is configured as shown. As a further example, elevation register 86 indicates the third bit position as having a bit value of "1". This indicates that master M3 is normally a low priority master and has undergone an interruption during its data transfer, and that it is to be elevated or promoted to the high priority ring. Each bit position of registers 84 and 86 are logically compared. The output of which is presented to a corresponding bit position within register 88. Given the expressed example, the third bit position of register 86 translates to a temporary high priority status at the third bit position of register 88. As the high priority ports of the high priority ring are polled, query is made as to all bit positions having a 1 value. If the query result is that no bit positions indicate a 1 value before arriving at the TO LP port 90, then query will begin on the low priority ring whereby 0 values are queried. Incrementing the appropriate pointers take place for each bus access. Eventually the third bit position, and therefore the corresponding port of the high priority ring, will be queried and the data transfer of the corresponding elevated, low priority device will be completed. Thereafter, that elevated device will be demoted by virtue of its corresponding bit value being changed to a 0 value within elevation register 86 and corresponding register 88.

CLAIMS:

1. A bus arbiter for a computer system, comprising:

a low priority arbitration ring having a plurality of low priority ports assigned to respective low priority peripheral devices;

a high priority arbitration ring having a plurality of high priority ports assigned to respective high priority peripheral devices and to a first one of the low priority peripheral devices elevated from the low priority ports during times in which data transfers to or from said first one of the low priority peripheral devices are interrupted; and

a high priority pointer which points to one of the high priority ports for prioritizing requests from among the high priority peripheral devices based on their proximity to the pointed-to high priority slot.

3. A bus arbiter for a computer system, comprising:

a low priority arbitration ring having a plurality of low priority ports assigned to respective low priority peripheral devices;

a high priority arbitration ring having a plurality of high priority ports assigned to respective high priority peripheral devices and to a first one of the low priority peripheral devices elevated from the low priority ports during times in which data transfers to or from said first one of the low priority peripheral devices are interrupted; and

a low priority pointer which points to one of the low priority ports for prioritizing requests from among the low priority peripheral devices based on their proximity to the pointed-to low priority slot.

5. A computer, comprising:

a microprocessor;

a plurality of high and low priority peripheral devices, at least one of which is coupled to a printed circuit board separate from another printed circuit board on which the microprocessor resides;

an arbiter coupled to a peripheral bus on which the plurality of high and low priority peripheral devices are operably coupled, the arbiter comprising:

a high priority arbitration ring having a plurality of high priority ports assigned to the plurality of high priority peripheral devices;

a low priority arbitration ring having a plurality of low priority ports assigned to the plurality of low priority peripheral devices;

wherein one of the low priority peripheral devices is assigned to one of the plurality of high priority ports and is granted mastership of the peripheral bus before any other of the low priority peripheral devices whenever a data transfer to or from said one of the plurality of low priority peripheral devices is retried; and;

a high priority pointer which points to one of the high priority ports for prioritizing requests from among the high priority peripheral devices based on their proximity to the pointed-to high priority port.

7. The computer as recited in claim 5, wherein each of the plurality of high priority ports is assigned to a respective one of the plurality of high priority peripheral devices.

9. The computer as recited in claim 5, wherein said one of the low priority peripheral devices is assigned to said one of the plurality of high priority ports if data transfer to or from said one of the plurality of low priority peripheral devices is interrupted.

10. The computer as recited in claim 5, wherein said one of the low priority peripheral devices is assigned to said one of the plurality of high priority ports if a target peripheral device adapted to receive data from said one of the plurality of low priority peripheral devices is unavailable.

13. The computer as recited in claim 12, further comprising logic coupled to combine the programmed value from the temporary elevation register with the programmed value within the configuration register to determine if the peripheral device corresponding to the programmed bit positions is assigned to the high priority peripheral devices or the low priority peripheral devices.

14. A computer, comprising:

a microprocessor;

a plurality of high and low priority peripheral devices, at least one of which is coupled to a printed circuit board separate from another printed circuit board on which the microprocessor resides;

an arbiter coupled to a peripheral bus on which the plurality of high and low priority peripheral devices are operably coupled, the arbiter comprising:

a high priority arbitration ring having a plurality of high priority ports assigned to the plurality of high priority peripheral devices;

a low priority arbitration ring having a plurality of low priority ports assigned to the plurality of low priority peripheral devices;

wherein one of the low priority peripheral devices is assigned to one of the plurality of high priority ports and is granted mastership of the peripheral bus before any other of the low priority peripheral devices whenever a data transfer to or from said one of the plurality of low priority peripheral devices is retried; and;

a low priority pointer which points to one of the low priority ports for prioritizing requests from among the low priority peripheral devices based on their proximity to the pointed-to low priority port.

16. The computer as recited in claim 10, wherein each of the plurality of high priority ports is assigned to a respective one of the plurality of high priority peripheral devices.

18. The computer as recited in claim 14, wherein said one of the low priority peripheral devices is assigned to said one of the plurality of high priority ports if data transfer to or from said one of the plurality of low priority peripheral devices is interrupted.

19. The computer as recited in claim 14, wherein said one of the low priority peripheral devices is assigned to said one of the plurality of high priority ports if a target peripheral device adapted to receive data from said one of the plurality of low priority peripheral devices is unavailable.

22. The computer as recited in claim 21, further comprising logic coupled to combine the programmed value from the temporary elevation register with the programmed value within the configuration register to determine if the peripheral device corresponding to the programmed bit positions is assigned to the high priority peripheral devices or the low priority peripheral devices.